VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

R Shiny - využití webového prostředí pro vizualizaci statistických dat

R Shiny - Web Framework for Statistical Data Visualization

2018 Tomáš Jirků

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Tomáš Jirků**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: R Shiny - využití webového prostředí pro vizualizaci statistických dat
R Shiny - Web Framework for Statistical Data Visualization

Jazyk vypracování: čeština

Zásady pro vypracování:

V dnešní době je kladen stále větší důraz na zpracování dat a jejich efektivní vizualizaci. V oblasti statistického zpracování dat je dnes velmi využívaným nástrojem jazyk R, který nabízí také webové prostředí pro následnou vizualizaci dat.
Cílem této práce je seznámení se s nástrojem R Shiny a vytvoření ukázkové webové aplikace.

1. Seznamte se s nástrojem R a prostředím R Shiny.
2. Popište možnosti a implemnetaci knihovny Shiny.
3. Navrhněte případovou studii v podobě webové aplikace nad specifickými daty, např. návštěvnost webu, lékařská data.
4. Implementujte webovou aplikaci.
5. Zhodnoťte možnosti využití prostředí Shiny.

Seznam doporučené odborné literatury:

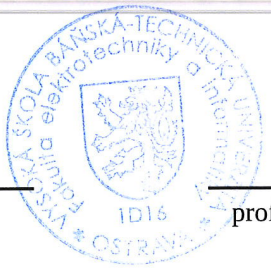
[1] Jon Duckett: JavaScript and JQuery: Interactive Front-End Web Development, Wiley, 2014, ISBN: 978-1118531648
[2] Sasha Vodnik: HTML5 and CSS3, Illustrated Complete, Course Technology, 2015, ISBN: 978-1305394049
[3] Erixc Elliot: Programming JavaScript Applications: Robust Web Architecture with Node, HTML5, and Modern JS Libraries, O'Reilly Media, 2014, ISBN: 978-1491950296
[4] Chris Beeley: Web Application Development with R using Shiny, Packt Publishing, 2016, ISBN: 978-1782174349

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Michal Radecký, Ph.D.**

Datum zadání: 01.09.2016

Datum odevzdání: 30.04.2018



doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 30.dubna 2018

Tomáš Jirků

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.

V Ostravě 30.dubna 2018

Tomáš Zirkl

Chtěl bych poděkovat vedoucímu své bakalářské práce Ing. Michalu Radeckému, Ph.D. Za odborné vedení, za pomoc a rady při zpracování této práce. Dále bych rád poděkoval své rodině za podporu během celého mého studia i při vypracování této práce.

Abstrakt

Tato bakalářská práce se zabývá vizualizaci dat v prostředí R Shiny. Na ukázku byla vytvořena webová aplikace, která využívá webového frameworku Shiny. V ukázkové aplikaci jsou využity také další knihovny pro práci s daty a grafikou.

Klíčová slova: R, Shiny, Webová aplikace, Framework

Abstract

This bachelor theses deals Data visualization in R Shiny. For demonstration it was created web application which use web framework Shiny. In sample app is used another libraries for work with data and graphics.

Key Words: R, Shiny, Web application, Framework

Obsah

Seznam použitých zkratek a symbolů **10**

Seznam obrázků **11**

Seznam tabulek **12**

1 Úvod **13**

2 Big data **14**

3 Technologie a nástroje pro zpracování rozsáhlých dat **16**
3.1 Ukládání dat . . . 16
3.2 Metody pro získání klíčových informací z dat . . . 18
3.3 Vizualizace dat . . . 19

4 Prostředí R **20**
4.1 R jako programovací jazyk . . . 20
4.2 Instalace platformy R . . . 20
4.3 Práce s konzolí . . . 20
4.4 Integrované grafické komponenty . . . 21
4.5 Import dat a ODBC . . . 22
4.6 Knihovny (balíky) . . . 23

5 Shiny **27**
5.1 Používané technologie . . . 27
5.2 Instalace a použití . . . 28
5.3 Shiny Server . . . 28
5.4 Struktura aplikace v Shiny . . . 28
5.5 Jak tvorba Shiny aplikací funguje . . . 30
5.6 HTML a CSS . . . 31
5.7 Shiny přidružené balíky . . . 32
5.8 Celkové zhodnocení balíku shiny . . . 34

6 Vývoj ukázkové webové aplikace **35**
6.1 Stručný popis webové aplikace . . . 35
6.2 Základní struktura ukázkového projektu . . . 35
6.3 Data . . . 37
6.4 Vizualizování dat a celkové dotváření obsahu . . . 38
6.5 Mrak slov (Word cloud) . . . 42

6.6 Ovládací prvky 43
6.7 Komplikace 45

7 Nasazení aplikace **46**

8 Závěr **47**

Literatura **48**

Přílohy **49**

A Seznam příloh **50**

Seznam použitých zkratek a symbolů

HTML	–	Hyper Text Markup Language
CSS	–	Cascading Style Sheets
MS-SQL	–	Microsoft Structured Query Language
ODBC	–	Open Database Connectivity
CRAN	–	Comprehensive R Archive Network

Seznam obrázků

1 3V vlastnosti, zdroj:www.bayometric.com 15
2 Metody data miningu, zdroj:http://vtm.e15.cz 18
3 Ukázka spuštěné console R pod windows 21
4 Archiv CRAN 24
5 Krabicový graf z balíku ggplot2, zdroj: https://bmscblog.files.wordpress.com . . . 25
6 Histogram z balíku plotly, zdroj: www.r-graph-gallery.com 26
7 Schéma shiny aplikace, zdroj: i1.wp.com 29
8 Použití vlastní javascriptové knihovny v Shiny, zdroj: https://docs.google.com . . 33
9 Náhled ukázkového projektu 35
10 Ukázka tabulky 39
11 Ukázka výsečového grafu 40
12 Svíčkový graf z balíku plotly 41
13 Ukázka WordCloudu 43
14 Kompletní přehled uživatelských vstupů z balíku shiny, zdroj: https://shiny.rstudio.com 44

Seznam tabulek

1 Seznam dostupných formátu pro export v R 22
2 Seznam Renderovacích funkcí v Shiny 31

1 Úvod

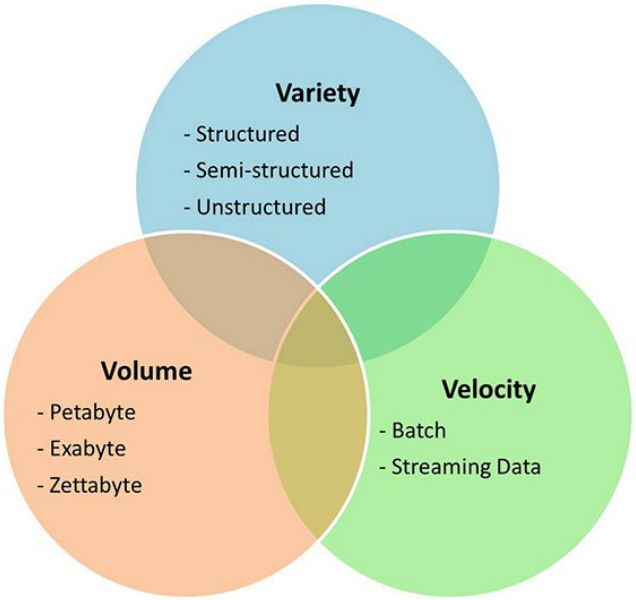
Žijeme ve světě informací. Kamkoliv se podíváme, všude nás obklopují informace ať už z televize, nebo internetu. S příchodem chytrých telefonů a vzrůstající oblibou mobilního internetu tento nárůst ještě zrychlil. Při používání těchto technologií, kdy můžeme online nakupovat, využívat internetové bankovnictví nebo prostě pouze vyhledávat informace, vytváříme data. Ačkoli se nám může zdát, že jsou tato data nedůležitá pro většinu marketingových společností, jsou nepostradatelná. Server The Economist označil data za novou ropu 21. století. Vzrůstající množství těchto dat nezadržitelně roste. Podle společnosti Cisco do roku 2019 vzroste objem dat, přenášených po internetu až pětinásobně. Za tímto velkým nárůstem má být vzrůstající obliba streamovaného videa a využívání sociálních sítí.[1] Pokud se ohlédneme do historie tak zjistíme, že se tato data změnila. Zatímco v minulých letech datoví analytici řešili zpracování převážně textových či numerických hodnot, tak dnes to jsou multimedia. Multimedia a různá nestrukturovaná data patří dnes k nejproblémovějšímu typu dat. Proto není divu, že vznikly různé nové pojmy a nástroje jako Big data nebo NoSQL databáze, které pomáhají tuto situaci řešit.

Tématem této bakalářské práce je objasnit základní pojmy týkající se Big dat a technologií pro jejich řešení. Tato práce má jako cíl seznámit čtenáře s vizualizací dat pomocí prostředí R a frameworku Shiny. V práci je kladen důraz na srozumitelnost a to i pro laika v prostředí R. Pro představu byl vytvořen ukázkový projekt, který má za cíl ukázat možnosti v prostředí R a Shiny včetně implementace různých druhů grafů. Projekt je zpracován jako webový dashboard, který kromě Shiny využívá i jiné knihovny pro práci s grafy a manipulaci s daty.

2 Big data

Pokud budeme hledat definici toho, co jsou to Big data brzo zjistíme, že neexistuje prozatím žádná jednoznačná ustálená definice. Obecně je možné říci, že Big data je termín, který se váže na soubory dat, jejichž velikost je natolik velká, že není možné data zpracovat běžnými databázovými nástroji nebo aplikacemi v rozumném čase. [2] Velikost ale není jediný problém těchto rozsáhlých dat. Všeobecně je možné tuto problematiku rozdělit do čtyř hlavních bodů. Tyto tři hlavní vlastnosti bývají označeny 3V od slov (volume, velocity, variety), přičemž o něco později byla přidána další vlastnost věrohodnost (veracity).

- Objem (volume): Velikostí či objemem dat se myslí data o velikosti desítek terabytů (TB) až po petabyty (PB). [3]
- Rychlost (velocity): Jak už bylo zmíněno, objem není největší problém těchto roszáhlých dat, ale spíše rychlost jejich vzniku. Pro představu si můžeme například uvést letoun Boeing, který za každých 30 minut vyprodukuje 10 TB dat. Tato data není nutné ani tak ukládat, ale spíše je efektivně zpracovat a získat důležité informace v reálném čase. Kromě těchto strojově vznikajících dat jsou dalším častým zdrojem sociální sítě. Například sociální síť Twitter má kolem 200 milionů uživatelů, kteří každý měsíc posílají kolem 400 milionů tweetu (kratkých komentářů). Popřípadě sociální síť Facebook má kolem 900 milionů uživatelů, kteří každý den sdílejí obsah a nahrají přes 300 milionů fotografií.
- Různorodost (variety): Různorodost je jeden z dalších problémů. Jedná se o semistrukturované data jako například *xml*, nebo zcela nestrukturované jako multimedia. Jelikož tato data nemají přesně danou strukturu, jako třeba v relačních databázích, není možné je tak jednoduše ukládat ve formě tabulek, atributů a pod. Tato multimedia jsou různé zvukové, video záznamy a obrázky, popřípadě i jiné grafické záznamy. Stojíme tedy před problémem vyhledávání v databázi multimedií jinak, než pomocí metadat nebo textových popisků.
- Věrohodnost (veracity): Jelikož se data často čerpají z veřejně přístupných zdrojů, nemusí být zcela věrohodné. Například získávání informací ze sociálních sítí, kde jsou tyto informace méně konzistentní a věrohodné oproti datům z databáze nějaké společnosti.



Obrázek 1: 3V

Analýza těchto dat může přinést cenné informace v komerčním i nekomerčním sektoru. V komerčním sektoru jsou to energetické, bankovní, telekomunikační nebo farmaceutické společnosti, shromažďující obrovská množství dat. Velcí giganti jako Google nebo Facebook tato data získávají při každé transakci mezi uživatelem a jejich systémem. Data jako jednotlivé záznamy nejsou příliš zajímavá, ale jako celek jsou cenná. Aby bylo možné z těchto dat získat klíčové informace, je potřeba použít některé z nástrojů a technologií, které jsou pro tyto rozsáhlá data určena.

3 Technologie a nástroje pro zpracování rozsáhlých dat

3.1 Ukládání dat

Při práci s rozsáhlými daty je klíčem k úspěchu efektivní ukládání. Proto ve světě začali vznikat různé komerční i nekomerční řešení, které by problematiku rozsáhlých dat měli řešit. Zde je nutné uvést, že není cílem za každou cenu používat novější technologie pro jejich ukládání. Zavedení těchto řešení by mělo probíhat v případě, že standardní relační databáze už nestačí. [12] V případě ukázkového projektu byla využita klasická relační databáze MS-SQL protože pro potřeby projektu byla dostačující.

3.1.1 NoSQL

NoSQL databáze tu s námi jsou již nějakou dobu, ale i tak jsou pro mnoho lidí velkou neznámou. Podle některých zdrojů by se mělo jednat o typ databázových systémů, silně se vymezujících proti SQL přístupu. Naopak některé zdroje uvádějí, že by název měl znamenat Not Only SQL. Tedy systémy, využívající i jiné metody přístupu k datům. Nicméně ať už název říká cokoli, všeobecně můžeme říci, že se jedná o typ databází, určených pro rozsáhlé množství dat.[2] NoSQL databáze jsou převážně nerelační, tedy jejich základní princip ukládání dat není postaven na tabulkách. Ačkoli by se mohlo zdát, že NoSQL databáze přišly pohřbít relační databáze, není tomu tak. NoSQL technologie přišla pouze jako varianta reagovat na jiné typy dat, se kterými si tradiční databázové systémy prostě neví rady. Tímto problémovým typem jsou nestrukturovaná data a převážně multimédia. O NoSQL databázích se mluví jako o systémech bez databázového modelu (schématu). V praxi samozdřejmě tyto databáze určité schéma ukládání mají, ale není přesně definováno, jako u relačních databází.

Mezi největší výhody NoSQL systémů patří:

- Škalovatelnost: Klasické databázové systémy používají vertikální škalovatelnost tzn. využívají výkonnější hardware. Zatímco databáze NoSQL využívají horizontální škalovatelnost, tím je myšleno, že úloha je rozdělená mezi množinu uzlů zvanou cluster, kde je možné vždy přidat další uzly.
- Datový model: NoSQL databáze nevyžadují žádné schéma, respektive v praxi data určité schéma mají, ale nemusí být přesně definováno jako např. u relačních databází. Jedná se spíše o nějaké meta data, a ne o nějaké konkrétní schéma.
- Cena Hardwaru: Výhodou NoSQL databázových systémů je, že využívají horizontální škalovatelnost, a díky tomu lze použít nepříliš drahé počítače.
- Rychlost čtení: Tento druh databázových systémů nabádá své uživatele k vytváření takových datových struktur, které budou co nejlépe odpovídat často pokládaným dotazům. Díky tomu je každý dotaz zpracován velmi rychle.

Tento typ databázových systémů často vzniká jako open-source nebo start-up projekty. To má sice i výhody, jako snadnou dostupnost pro každého, jenže má to také řadu nevýhod.

Jedna z nich je odlišný přístup k datům v rámci rozdílných distribucí. Administrace je tedy mnohem složitější, oproti standardním relačním databázím, kde je jednotný přístup k datům pomocí SQL dotazování.

Databází tohoto typu je spousta, nicméně pro představu budou uvedeny dvě nejznámější distribuce:

- MongoDB: Patří k nejznámějším NoSQL databázovým systémům. Podle tvůrců jde o dokumentově orientovaný typ databáze. Tedy každý záznam je brán jako dokument. Každý dokument má (id) a dokumentový typ. Záznamy jsou soubory podobné formátu JSON (MongoDb je nazval BSON). Je to specifikace JSONu, která je převedena do binární podoby. Tyto struktury nemají žádný předem připravený formát, takže každý záznam může mít jiné položky. K dotazování má MongoDB připraven vlastní dotazovací jazyk. Tento jazyk připomíná jazyk SQL. Pro získání dat z databáze je možné také použít přímo javascriptovou metodu, spuštěnou na straně serveru. K hromadným operacím s daty je v MongoDB k dispozici MapReduce. Jedná se o programovací model pro zpracování velkých objemů dat pomocí paralelního zpracování. [8]

- Redis: Je další varianta NoSQL databází. Ukládání dat je založeno na key-value principu, tím je myšlen systém, kdy se pod jeden klíč uloží jedna hodnota. V případě Redisu je to pod jeden klíč ukládání několik datových struktur. Redis funguje na principu in-memory a to znamená, že si data primárně uchovává v paměti. Z toho plyne vysoká rychlost čtení dat.

  Přestože data jsou primárně uložena v operační paměti, je možné data ukládat také na pevný disk. Pokud dojde k selhání, lze obnovit databázi do posledního známého stavu. [16]

3.1.2 In-memory databáze

O tomto tématu byla zmínka již v předchozím textu, zabývajícím se databázovým systémem Redis. Jedná se o druh databází, uchovávající data v paměti počítače, bez nutnosti přístupu k pomalejším datovým uložištím. To umožňuje mnohem rychlejší provádění transakcí a databáze jsou tak mnohem více efektivní.

Tuto metodu využívají jak NoSQL, tak SQL systémy. In-memory přístup implementovaly i známe společnosti jako Oracle nebo Microsoft.[10]

3.1.3 Distribuované uložiště

Distribuované uložiště je koncept, využívající více serverů pro uložení dat s tím, že se chovají jako jeden systém. Datová centra mohou být na různých místech světa a přesto se uživateli jeví, že má data na jednom místě. Tato metoda ukládání má celou řadou výhod. Například umožňuje

škálovatelnost tedy přidávat další servery, čímž se zvyšuje výkon i prostor pro ukládání. Navíc lze využít klasické servery a spojit je do jednoho celku, bez nutnosti velkých cenových nákladů. [9]

3.2 Metody pro získání klíčových informací z dat

Kromě efektivního ukládání dat je potřeba také data správně zpracovat a získat z nich klíčové informace. Slouží k tomu různé statistické a matematické metody. Tyto metody se často používají společně a dali tak vzniknout novým pojmům a směrům, jak s daty pracovat.

S problematikou Veledat se často skloňují dva pojmy: Data mining (dolování dat) a Predikativní analýza.

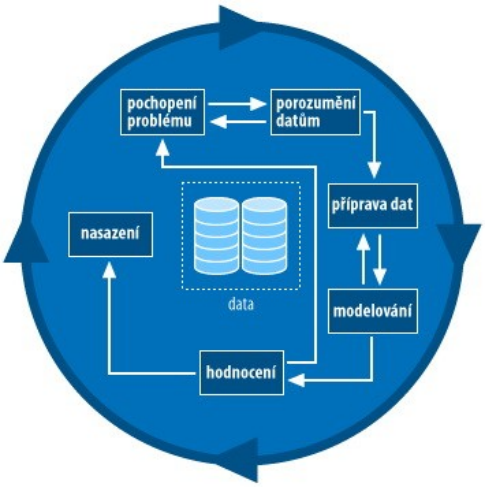
3.2.1 Data mining

Data mining nebo také dolování dat je metoda, kdy se pomocí různých matematických funkcí a statistiky hledají ve velkém objemu dat skryté informace. [13] Nejčastěji se používá v marketingu, ale také i ve vědecké oblasti, například v bioinformatice.

Datový analytici, kteří mají data na starost, dokáží z databáze klientů mobilního operátora získat informace o posledním půl roce klienta, který poté odešel ke konkurenci a jsou schopni najít tak i jiné klienty s podobným vzorcem chování. [14]

Datamining slouží také různým obchodníkům. Ze získaných informací o koupi lze vytvořit skupinu zákazníků, kterým je poté nabídnut určitý produkt, kde je větší šance, že si ho koupí oproti jiné skupině.

Co se týče bioinformatiky, zde se data mining používá pro získání skrytých informací z genomu. Používá se třeba pro získání informace z DNA a určení, jestli daný člověk může mít předpoklad pro nějakou závažnou nemoc.



Obrázek 2: Metody data miningu podle určitých fází

3.2.2 Prediktivní analýza

Tento druh analýzy spojuje již některé používané nástroje jako různé druhy statistické analýzy, data mining, prediktivní modelování, textové analýzy, analýzy entit, vyhodnocování v reálném čase, strojové učení a další.

Podstata této analýzy je využití starších dat pro předpověď budoucího vývoje.[15] Tato analýza se využívá pro zpracování nejen strukturovaných dat, ale i textů a obrázků.

3.3 Vizualizace dat

Při práci s velkým objemem dat je žádoucí data vizualizovat. Vizualizovaná data jsou lépe čitelná a umožňují rozpoznat dosud skryté souvislosti, které bychom jinak snadno přehlédli. Při vizualizaci dat nejsou důležité detaily, ale spíše vztahy a celkový náhled, aniž by jsme data nějak složitě studovali.

Při vizualizaci je nutné si položit některé základní otázky: Co v datech hledáme ? A proč ? Výsledky po vizualizacích by měli být co možná nejstručnější a neměli by trpět žádnými rušivými vlivy. Často jsou informace potlačeny na úkor designu. Takové vizualizace často lákají pouze na grafický vzhled, ale nesplňují základní princip a to sdělit klíčové informace.[17]

3.3.1 Druhy vizualizací

Aby jsme mohli data vizualizovat, nemusíme používat pouze grafy, ale existují také jiné možnosti zobrazení.[11]

- Lineární: Tabulky nebo seznamy položek, uspořádány jedním znakem (např. abecedně).
- 2D zejména geoprostorové: Kartogramy,mapy proporcionálních bodových znaků,obrysové mapy.
- 3D: 3D počítačové modely, počítačové simulace.
- Temporální: Grafy časových řad, obloukové diagramy, korelační diagramy.
- Multidimenzionální: Koláčové grafy, histogramy, sloupcové grafy, stromové mapy, tag clouds (oblaka štítků), graf pavouk.
- Stromové / Hierarchické: dendrogramy, radiální stromové grafy, hyperbolické stromové grafy.

4 Prostředí R

Platforma R je programovací jazyk a výpočetní prostředí pro rozsáhlé operace s daty. Umožňuje pokročilou analýzu dat, včetně vytváření různých grafických výstupů. Lze provádět jakékoli statistické operace jako regrese, analýzy, rozptyly až po nějaké specializované témata, třeba analýzy časových řad. Platforma R se používá v různých vědeckých odvětvích v lékařství, chemii, fyzice, ekonomii a v mnoha dalších. O prostředí R se také často mluví jako o jednom z hlavních nástrojů, používaných v problematice rozsáhlých dat. Výhoda prostředí R je také v dostupnosti. Je zcela zdarma za podmínek GNU (General Public Lincence). Navíc je také multiplatformní. Můžeme ho nainstalovat na operačních systémech Windows, Unix, Linuxové distribuce a MAC OSX.[7]

4.1 R jako programovací jazyk

Platforma R slouží také jako programovací jazyk. Spadá do rodiny interpretovacích jazyků. To jsou jazyky, které pro svou činnost potřebují interpret. Interpret je software, který umožňuje provádění zdrojového kódu. Opakem jsou kompilované jazyky. Tedy jazyky, kde se zdrojový kód musí přeložit pomocí překladače do strojového kódu. Jazyk R vychází z jazyka S, který vyvinul profesor John Chambers a jeho kolegové v Bellových laboratořích. Oba jazyky jsou rozdílné, ale i tak je pravděpodobné, že kód napsaný v S bude fungovat bez větších zasahů také v R.[4] Jazyk R je (case sensitive), tím myslíme, že je citlivý na psaní velkých a malých písmen.

R podporuje procedurální programování s tím, že je také částečně objektový. OOP je v jazyce R reprezentováno generickými funkcemi. Například funkce *print()* je generická funkce, která umožňuje vytisknout prakticky jakýkoli objekt.

4.2 Instalace platformy R

Software R je možné zdarma stáhnout z oficiálního webu `https://cloud.r-project.org/`. Software je možné nainstalovat na operační systémy: Windows, Mac a Linuxové distribuce (Ubuntu, Redhat, Debian, Suse). Na Linuxech lze pracovat s R přímo v terminálu, kde stačí zadat klíčové slovo R a běhové prostředí je připraveno k práci.

Pod Windows můžeme spustit R v jeho consolové formě. Kromě toho už existují také některé vývojové prostředí (IDE).

Nejznámější pro práci v R jsou : R Studio, Visual Studio for R, Eclipse.

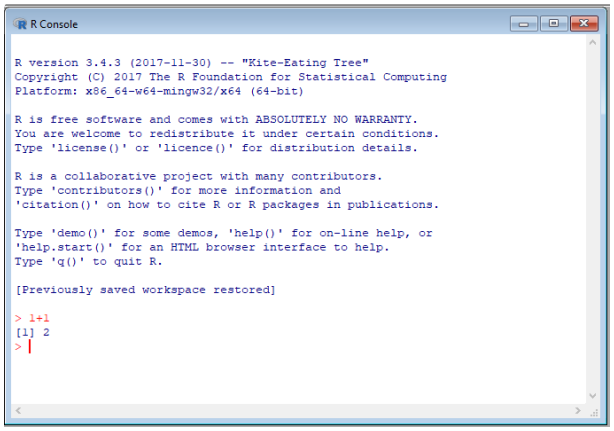
4.3 Práce s konzolí

Okno s konzolí je základní rozhraní platformy R. Konzole slouží jako vypočetní prostředí a lze do ní vkládat také programové konstrukce. Pokud konzoli spouštíme úplně poprvé, všimněme si znaku '>'. Tento znak označuje vstupní řádek konzole. Vstupní řádky jsou označeny červenou barvou a výstupní řádky barvou modrou. Napíšeme-li do konzole nějaký jednoduchý příkaz,

třeba 1+1 a stiskneme klávesu Enter, dostaneme [1] 2. Tato operace je vlastně součet dvou jednoprvkových vektorů a výsledkem je také jednoprvkový vektor. Výhodou je, že konzole má historii příkazů. Kdyby jsme napsali příkaz špatně, nemusíme ho znovu celý psát, ale vrátíme se k němu šipkou ↑.

Příkazy se nemusejí psát pouze na jeden řádek. Jakmile by jsme příkaz rozdělili na několik řádků, můžeme si v konzoli všimnout znaku '+'. Tím konzole říká, že čeká na ukončení příkazu.

Pokud už nechceme pokračovat , můžeme napsat příkaz *q()*, pro ukončení práce s konzolí.



Obrázek 3: Ukázka spuštěné console R pod windows

4.4 Integrované grafické komponenty

Pro práci s grafikou má platforma R tři klíčové komponenty jsou to:

grDevices - Je knihovna pro práci s grafikou, dostupná v základní instalaci softwaru. Tato knihovna je vlastně grafický engine celé platformy R.[19] Je zodpovědná za vykreslování veškerých grafických objektů včetně fontů a barev. Při standardní instalaci se nám tato knihovna klasicky spustí. Pokud máme z nějakého důvodu nestandardní instalaci, je možné knihovnu načíst pomocí *library(grDevices)*.

Další komponentou pro práci s grafikou je **graphics**. Je to knihovna, která umožňuje vytváření samotných grafů. Klíčovou položkou knihovny je obecná generická funkce *plot()*. Funkce *plot*, je velice univerzální a pomocí ní lze vytvořit spoustu (X,Y) grafů s body a čárami. Obsahuje například metodu *plot.histogram*, která umožňuje vytvoření histogramu pomocí *hist()*. Kromě toho je možné vytvořit nasledující typy grafů:

1. Čárové grafy (Line charts) *lines()*.
2. Sloupcové grafy (Bar charts) *barplot()*.
3. Histogramy (Histograms) *hist()*.
4. Koláčové grafy (Pie charts) *pie()*.

5. Bodové grafy (Dot charts) *dotchart()*.

Grid- Je další velice důležitá součást pro práci s grafikou. Tato komponenta neobsahuje grafy, ale obsahuje nástroje pro jejich vylepšení. Knihovna má integrovány funkce pro editování, změnu velikosti a mazání určitých objektů z grafů a mnoho dalších funkcí.

4.4.1 Grafické formáty

Jakmile vytvoříme nějaký graf obvykle s ním chceme dále pracovat. Pokud ho nebudeme zobrazovat přímo přes webové rozhraní pomocí platformy R. Nabízí se možnost graf uložit do nějakého určitého formátu. Tím formátem mohou být různé obrázkové formáty, nebo nějaký typ dokumentu. Za vyexportování grafu zodpovídá knihovna *grDevices*. Pro každý typ formátu knihovna *grDevices* obsahuje specifickou funkci.

Následující tabulka zobrazuje podporované formáty a funkce pro exportování do těchto formátů.

formát	funkce
PNG	png()
JPEG	jpeg()
TIFF	tiff()
BMP	bmp()
Adobe PDF	pdf()
Adobe Postscript	postscript()
SVG	svg()

Tabulka 1: Seznam dostupných formátu pro export v R

4.5 Import dat a ODBC

Než začneme s daty pracovat, je nutné si rozmyslet, jak data do prostředí R načteme. Můžeme například využít nějaký typ datového souboru, nebo data načteme z databáze. Je možné pracovat jak s SQL, tak s NoSQL databázemi. Protože databázových distribucí je velká spousta, bylo by vhodné využít technologie ODBC.

4.5.1 ODBC

Open Database Connectivity je softwarové API pro přístup k databázovým systémům. Cílem ODBC je nezávislost na databázovém systému, operačním systému, popřípadě programovacím jazyku. K použití ODBC je potřeba mít nainstalován vhodný driver, který slouží jako most mezi aplikací a databázovým systémem.[18]

Model ODBC se skládá ze tří hlavních částí

- Klient
- Databázový server
- ovladač ODBC

Platforma R využívá tuto technologii jako jednu z možností pro přístup k datům. Tradičně ale tato funkcionalita není dostupná a je nutné stáhnout rozšíření v podobě knihovny. Knihovna se jmenuje *RODBC* a práce s ní je demonstrována v podkapitole 6.3.1.

4.5.2 Import dat ze souboru

Kromě čerpání dat z databáze, je možné data získat také ze souboru. Ideální je použít formát souboru **CSV**, se kterým se v prostředí R velmi jednoduše pracuje. CSV je souborový formát pro ukládání tabulkových dat. Hodnoty v souboru jsou odděleny čárkou, popřípadě středníkem. Tento typ souboru se hojně využívá pro jeho jednoduchost. Pro práci s ním lze v prostředí R využít funkce *read.csv()*, nebo *read.table()*. Při práci s těmito funkcemi je potřeba zadat cestu k souboru. Buď zadáme celou cestu, nebo využijeme pracovní adresář. Cestu k adresáři zjistíme příkazem *getwd()*. Poté už stačí pouze soubor umístit do pracovního adresáře a jako parametr ve funkcích zadat jméno souboru včetně přípony.

Kromě jména souboru je nutné zadat i další klíčové parametry jako:

- Header - logická hodnota v závislosti na tom, jestli je první řádek název sloupců.
- Sep - je textový znak, který říká, čím jsou v souboru odděleny jednotlivé hodnoty.

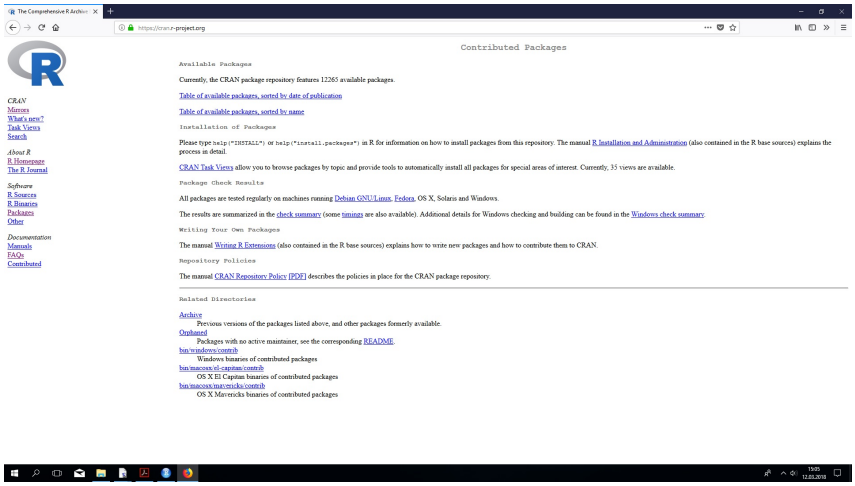
4.6 Knihovny (balíky)

Knihovny v prostředí R jsou důležitá součást jazyka. Pomáhají rozšiřovat možnosti jazyka a umožňují komunitě vývojářů přidávat funkcionalitu, která v základním softwaru chybí. Knihovnám v jazyce R se říká balíky packages. Tyto balíky můžeme stáhnout z archívu CRAN popřípadě i z jiných zdrojů jako github.com. Tyto balíky jsou obvykle zaměřeny na určitou problematiku, jako třeba rozšíření množství grafů nebo balík pro rozsáhlejší manipulaci s daty a jiné.

Kromě toho má v sobě software R již několik předpřipravených balíků, které získáme při základní instalaci. Balíky packages jsou vytvořeny obvykle v c/c++ popřípadě přímo v jazyce R.

4.6.1 Archiv CRAN

Comprehensive R Archive Network (CRAN) je hlavní úložiště pro balíky v R. Počet těchto knihoven neustále roste. Ke dni 12.3.2018 je v archivu CRAN dostupných 12 265 balíků.



Obrázek 4: Archiv CRAN

4.6.2 Instalace a použití balíku

Pokud bychom chtěli instalovat balík dostupný v archivu CRAN. Pro instalaci se používá příkaz *install.packages*, kde jako parametr je název balíku. Pokud bychom například chtěli nainstalovat balík *ggplot2*, nainstalujeme ho *install.packages("ggplot2")*. Abychom mohli balík začít používat, musíme v kódu specifikovat, který balík potřebujeme. K tomuto účelu použijeme příkaz *library*, kde jako parametr je opět název balíku. Například pro balík *ggplot2*, by to bylo *library("ggplot2")*. Jakmile takto balík načteme, máme již veškerou funkcionalitu balíku dostupnou.

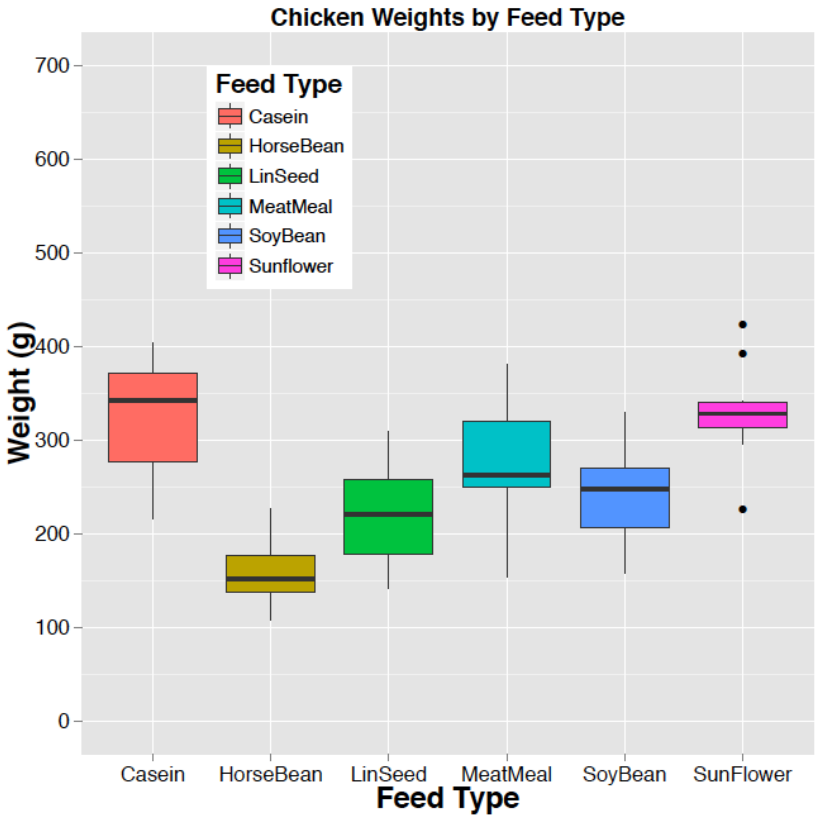
4.6.3 Ggplot2

Rozšíření *ggplot2* je balík pro vytváření grafů v prostředí R. Obsahuje velkou zásobu různých typů grafů od standardních grafů, jako bar chart až po méně známé grafy, jako text label chart. Syntaxe vytváření grafu se od základních grafů v R poněkud liší. V *ggplot2* se graf vytváří pomocí jednotlivých grafických komponent, kde se jednotlivé komponenty spojují pomocí znaménka '+'. Jednotlivými částmi mohou být popisky, legendy a pak také samotné druhy grafů.

Základem vytváření grafů pod *ggplot2* jsou dvě funkce:

1. ggplot
2. qplot

Obě funkce slouží pro vytváření grafů. Všeobecně je ale možné říci, že *ggplot()* se používá pro vytváření složitějších grafů. Zatímco *qplot()* se využije tam, kde se zobrazuje jednodušší množina dat.

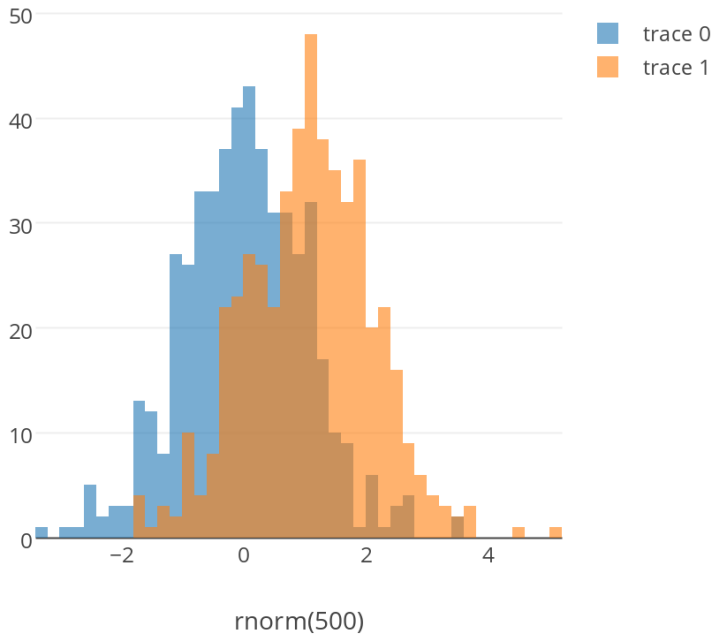


Obrázek 5: Krabicový graf z balíku ggplot2

4.6.4 Plotly

Plotly je bezplatná open-source knihovna pro vizualizaci dat. Tato knihovna využívá technologie D3 a WebGL pro vytváření grafických výstupů. Kromě použití v R je *plotly* dostupné i pro jiné jazyky, jako Python a Javascript. Zatímco grafy v R jsou obvykle renderovány jako obrázky, *plotly* grafy jsou serializované json soubory.

Plotly umožňuje vytvářet grafy dvěma způsoby. První z nich je pomocí funkce *plot_ly ()*, zde si vybereme požadovaný typ grafů. Druhá možnost je transformovat *ggplot* graf na objekt v *plotly*, k tomu je dostupná funkce *ggplotly()*. Nabízí se otázka, proč vytvářet graf v *plotly*, zatímco se dá vytvořit v *ggplot2*. Výhoda *plotly* je v interaktivitě. Grafické výstupy jsou velice interaktivní, je možné v grafech zoomovat, posunovat kurzorem a jiné věci. Navíc v případě využití grafu na webovém rozhraní jsou grafy velice responzivní.



Obrázek 6: Histogram z balíku plotly

4.6.5 Dplyr

Je balík pro manipulaci s daty. Obsahuje důležité funkce pro práci s daty jako filtrace, agregace, výběr určitých sloupců (select) a také vytváření nových sloupců. Jelikož dplyr obsahuje prakticky vše jako SQL dotazování, je možné si načíst do prostředí R celou tabulku a filtrovat data až v prostředí R, místo neustálých dotazů na databázový server.

4.6.6 Tm (Text Mining Package)

Tm - je balík pro text mining, který umožňuje získat klíčová slova z textu. Obsahuje různé funkce pro odfiltrování, počítání duplicitních slov a také databázi slov, které nemají hlubší význam.

5 Shiny

Shiny je open-source balík, který umožňuje vytváření interaktivních webových aplikací přímo v prostředí R. Balík *shiny* neslouží přímo pro vizualizaci, ale umožňuje ji zobrazovat přes webové rozhraní.

Shiny aplikace lze vytvářet prakticky bez znalostí HTML, CSS nebo Javascriptu. Tím je myšleno, že je možné vytvářet aplikace pomocí předem připravených šablon, ale je zde možnost vytvořit si i vlastní šablony. Aplikace se vytváří pomocí propracovaných vstupů a výstupů, které dotvářejí vysokou interaktivitu aplikace. V případě potřeby je možné také využít html tagy, které jsou do *shiny* integrovány.

5.1 Používané technologie

Shiny používá pro svou základní funkčnost dvě poměrně známé knihovny. Jsou to knihovny Bootstrap a jQuery.

5.1.1 Bootstrap

Bootstrap je CSS a javascriptový framework, který vznikl ve společnosti Twitter. Zaměstnanci Twitteru řešili neustále problémy s nekonzistencí mezi interními nástroji a rozdíly mezi vzhledy v jednotlivých stylech. Proto později dostali za úkol vytvořit jednotný (prozatím interní) nástroj, který by veškeré tyto technologie sjednotil.

Firma později tento CSS framework uvolnila jako open-source. Je možné ho tedy použít i ke komerčním účelům.

K hlavním výhodám používání frameworku Bootstrap patří:

- Responzivita - Je klíčová vlastnost frameworku Bootstrap. Umožňuje přizpůsobit stránku rozlišení zařízení, na kterém web prohlížíme. Tato vlastnost se stala klíčovou s příchodem mobilních telefonů.
- Mobile first - Tím jak rostlo množství uživatelů, prohlížejících webové stránky z mobilu, bylo potřeba framework přepsat tak, aby podporoval technologie, které pomáhali tyto stránky lépe zobrazovat na mobilních telefonech.

5.1.2 JQuery

JQuery je knihovna, která má usnadnit práci s javascriptem. Jedná se o nejrozšířenější javascriptovou knihovnu. Je multiplatformní, (funguje na více operačních systémech). A rovněž je také open-source, lze ji tedy použít i ke komerčním účelům.

Jedna z výhod oproti standardnímu javascriptu je úspora kódu. Kód v jQuery je obvykle úspornější a přehlednější.

Framework je možné použít pro:

- Manipulaci s objekty DOM.
- Manipulaci s CSS.
- Události.
- Animace (poměrně jednoduchá tvorba).
- Efekty (předem definované funkce).

Tento framework buďto můžeme jednoduše stáhnout a poté přidat k našim stránkám, nebo je také možné používat framework online a nalinkovat ho např. prostřednictvím googlu.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"
    ></script>
```

Výpis 1: Použití knihovny jQuery ze zdroje google

5.2 Instalace a použití

Ačkoli je *shiny* velice propracovaný, je to stále pouze knihovna platformy R. Tudíž instalace *shiny* není nijak extra odlišná oproti instalaci jiných balíků. Pro instalaci *shiny* použijeme příkaz *install.packages("shiny")*. Jakmile je balík nainstalován, stačí v nově vytvořeném projektu pouze definovat příkazem *library("shiny")*, že chceme používat tento balík.

5.3 Shiny Server

Shiny Server je speciálně navržený software, který umožňuje hostování Shiny aplikací. Díky tomu mohou být dostupné Shiny aplikace odkudkoli. Tento software si můžeme nainstalovat na vlastní server a mít tak plnou kontrolu nad celou aplikací. Každá Shiny aplikace je hostována na vlastní webové adrese. Pokud uživatel tuto adresu navštíví, aplikace se automaticky spustí. Jakmile uživatel skončí práci a odejde, Shiny Server tuto aplikaci automaticky zastaví.

Software Shiny Server je zdarma dostupný na adrese : `https://www.rstudio.com/products/shiny/download-server/`. Software můžeme nainstalovat na Linuxových platformách: Ubuntu, RedHat/CentOS a Suse. Operační systémy jako Windows nebo Mac OSX prozatím nejsou podporovány.

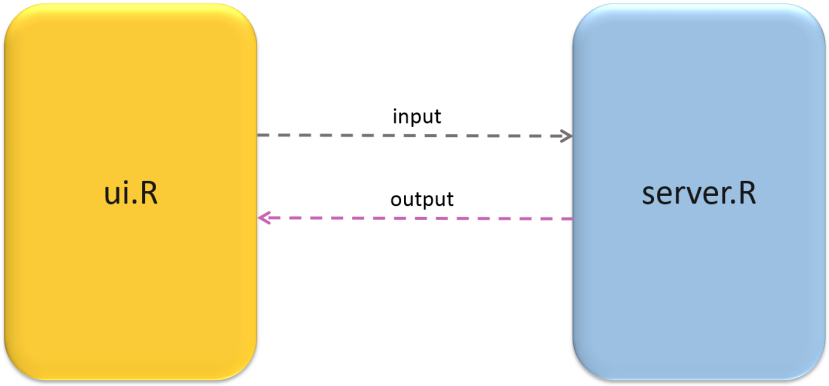
5.4 Struktura aplikace v Shiny

Aplikace v Shiny se vytvářejí pomocí jazyka R, kde se využívají funkce integrované v balíku shiny. Aplikace je tvořena ze dvou částí - uživatelského rozhraní a serverové části.

Pro vytvoření aplikace si musíme vytvořit dva soubory *ui.R* a *server.R*, kde *ui.R* reprezentuje část uživatelského rozhraní a *server.R* serverovou část.

Uživatelské rozhraní *ui.R* - Tato část aplikace obsahuje veškerý kód, týkající se vzhledu a vstupu uživatele. Stručně řečeno *ui.R* obsahuje vše, co vidí koncoví uživatelé. Definuje se zde, jak aplikace bude vypadat. Je možné zde například vytvořit menu, přidávat textový nebo grafický obsah v podobě obrázků, grafů a videí.

Serverová část server.R - Zde je obsažena logická část aplikace. Je to veškerá část kódu, která uživateli zůstává utajena. Provádí se tady veškerá funkční část, včetně operací s daty, od získání dat až po předání a zpracovaní dat. Je zde obsažen také veškerý kód, týkající se renderování, ať už grafů nebo třeba tabulek a jiných objektů.



Obrázek 7: Schéma shiny aplikace

Kromě rozdělení aplikace na dva soubory je od verze 0.10.2 možné celou aplikaci uložit do jednoho souboru *app.R*. Model rozdělení aplikace na dvě části je zde také přítomen, jediný rozdíl je v tom, že je vše uloženo do jednoho souboru.

```
library(shiny)

ui <- fluidPage(
  titlePanel("bp Shiny"),
     sidebarLayout(
        sidebarPanel(),
    mainPanel()
  )
)

server <- function(input, output) {
}
shinyApp(ui = ui, server = server)
```

Výpis 2: Zakladni tělo Shiny aplikace

5.5 Jak tvorba Shiny aplikací funguje

Shiny aplikace se vytváří poměrně zajímavým způsobem. Zatímco v klasických skriptech je prováděn kód řádek po řádku, v Shiny je situace poněkud odlišná. Jak už bylo popsáno v kapitole 5.4 Shiny aplikace se skladá ze dvou částí: serverové a části uživatelského rozhraní. V serverové části aplikace vytváříme bloky kódu výstupy (outputs) , které reagují na ovládací prvky aplikace vstupy (inputs). Tyto bloky kódu jsou v nečinnosti, dokud uživatel nezačne pracovat s těmito ovládacími prvky. V takovém případě se blok kódu v serverové části aktivuje a provede. Zní to celkem jednoduše, ale mohou nastat určité problémy. Pokud bychom měli dva výstupy, závislé na jednom ovládacím prvku, který výstup se provede jako první ? Ideální by bylo, aby se provedly oba současně, ale v praxi se provede vždy jeden rychleji. Navíc pořadí provedení těchto výstupů je náhodné. [5] To může způsobit určité kompikace, se kterými programátor předem nemusí počítat. Naštěstí je zde také řešení a to v podobě reaktivity.

5.5.1 Reaktivita

Pro tvorbu interaktivních aplikací je potřeba pochopit model reaktivního programování použitého v Shiny. Částečně byl tento model nastíněn už v podkapitole 5.5.

V Shiny jsou definovány tři druhy objektů pro reaktivní programování.[6] Tyto objekty jsou nazvány:

- reaktivní zdroje.
- reaktivní vodiče.
- reaktivní koncové body.

Reaktivním zdrojem je myšlen obvykle nějaký ovládací prvek (input), se kterým uživatel pracuje přes rozhraní prohlížeče, může to být textbox, tlačítko a nějaké další prvky třeba přepínač. Reaktivním koncovým bodem je obvykle nějaký objekt (output) v prohlížeči uživatele. V případě Shiny je to nejčastěji graf nebo nějaká tabulka hodnot. Pro představu, pokud bychom měli např. textbox, pomocí kterého by jsme filtrovali hodnoty v tabulce.

Kdykoli by jsme provedli pomocí ovládacího prvku nějakou změnu, v případě textboxu zadání hodnoty, tabulka by na toto musela reagovat. Posledním reaktivním typem je vodič. Ten se využívá pro zapouzdření pomalých nebo časové náročných operací. Ve většině případů nám stačí použít pouze dva zmíněné objekty, ale někdy se nevyhneme ani poslednímu typu reaktivních objektů. Ten byl pro větší pochopení implementován v ukázkovém projektu a práce s ním je popsána v podkapitole 6.6.1

5.5.2 Funkce render

Funkce render se používají pro vytváření různých druhů objektů v serverové části aplikace. Nejčastěji se v Shiny renderují grafy, ale je možné použít render funkci i pro vykreslení obrázku, tabulky, textu, html kódu a ovldacích prvků .

Aby bylo jasné, kde se má objekt vytvořit, musí se v uživatelském rozhraní (UI) definovat výstup (Output). Shiny pro tyto požadavky obsahuje funkce. Každá funkce slouží pro vyrenderování určitého typu objektu.

Render	Output	výstup
renderPlot	plotOutput	Graf
renderImage	imageOutput	Obrázek
renderUI	uiOutput	UI element
renderTable	tableOutput	Tabulka
renderText	textOutput	Text
renderDataTable	dataTableOutput	Interaktivní tabulka

Tabulka 2: Seznam Renderovacích funkcí v Shiny

5.6 HTML a CSS

HTML - HyperText Markup Language neboli značkovací jazyk, využívající hypertextové odkazy, který se používá pro tvorbu webových stránek.

Současně je ve verzi 5, která přinesla některé novinky jako přehrávání multimedia ve webovém prohlížeči, bez nutnosti použití flash playeru.

Přibyly také nové tágy jako *<header> <footer>* a další, které pomáhají vytvářet základní části stránky, bez nutnosti použití tagu * *.

Kaskádové styly (v anglickém jazyce Cascading Style Sheets) zkráceně CSS je technologie, která odděluje obsah webové stránky od vzhledu. Pomocí CSS lze jednoduše definovat druhy písma, barvu pozadí stránky, zarovnání, rozlišení a další vlastnosti elementů. Výhoda kaskádových stylů je, že tyto deklarované vlastnosti mohou sdílet všechny stránky celého webu. Základem CSS jsou selektory a deklarace.

Selektory jsou jednotlivé html elementy např. *body* nebo *h1*, přičemž deklarace se skládá ze dvou částí, z vlastnosti třeba *color* a hodnoty např. *blue*.

Tyto deklarace se zapisují do hranatých závorek jako bloky kódů a jsou odděleny středníky.

5.6.1 HTML a CSS v Shiny

Chceme -li využívat HTML přímo v prostředí Shiny, využijeme k tomu funkci *tags*, kde za znakem $ specifikujeme, jaký HTML tag potřebujeme. Například budeme chtít udělat nadpis *h3*, napíšeme tags$h3("Nadpis").

Seznam dostupných tagů zjistíme příkazem *names(tags)*. V současné době je dostupných až 110 tagů.

Kromě psaní html značek přímo v samotném Shiny, můžeme také přidávat větší bloky kódu napsané v HTML. Tento HTML kód ale musí být oddělen od samotného Shiny. Je k tomu dostupná funkce *HTML()*, kdy celý blok kódu uzavřeme do uvozovek.

Následující ukázka demonstruje přidávání html kódu do Shiny.

```
ui <- fluidPage(
  HTML("
    <h1>Bar chart</h1>
    <p>sloupcovy graf v shiny</p>"),
```

Výpis 3: Použití html v shiny

Kromě HTML můžeme samozdřejmě využít také Kaskádové styly. Jestliže potřebujeme měnit například barvu a velikost textu nebo nějaké jiné přednastavené vlastnosti v Shiny, CSS nám to umožní. Pro práci s Kaskádovými styly je ideální si vytvořit vlastní soubor, kam styly budeme ukládat. Tento soubor ale není možné uložit kdekoli. Musí být umístěn přesně v adresáři www.

Tento adresář slouží pro soubory definované tvůrcem aplikace. Do adresáře www se kromě css souborů ukládají také javascriptové soubory a obrázky. Jakmile máme soubor vytvořen, musíme pouze nalinkovat daný css soubor v uživatelském rozhraní *ui.R*. Soubor nalinkujeme pomocí *tags$head* a *tags$link*. Kompletní ukázka nalinkování souboru je demonstrování v podkapitole 6.2.2.

5.7 Shiny přidružené balíky

Kromě základního balíku *shiny* existují i další, které s ním spolupracují. Existují další menší balíky, které dále rozšiřují jeho možnosti nebo je vylepšují. V této podkapitole budou některé zmíněny.

5.7.1 ShinyFiles

Tento balík rozšiřuje možnosti práce se soubory v prostředí Shiny. Obsahuje funkce pro získání informací o souboru před jeho odesláním do Shiny. Dále obsahuje filtry, které umožňují načíst pouze soubory určitého typu.

5.7.2 Shinycssloaders

Velmi malý balík, který obsahuje několik animací pro efekt načítání souborů, grafů, tabulek a dalších prvků.

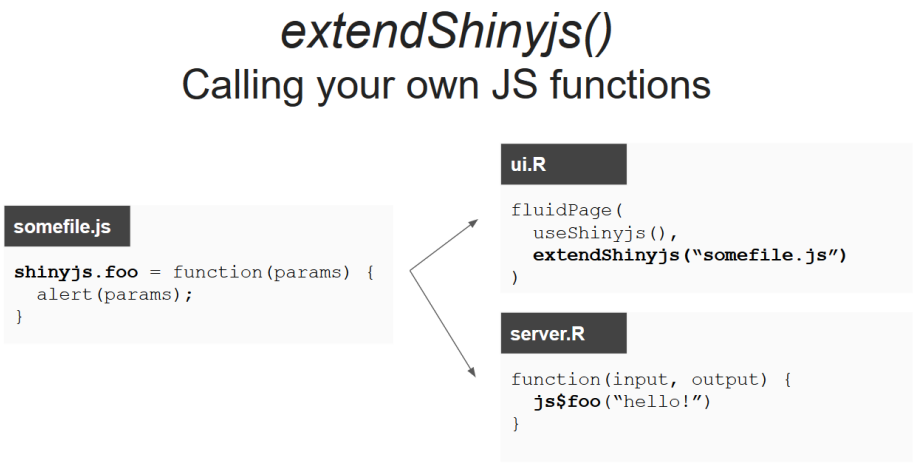
5.7.3 Shiny.semantic

Shiny semantic rozšiřuje grafické možnosti uživatelského rozhraní. Obsahuje různé druhy šablon, se kterými se dá pracovat přímo v R, bez nutnosti použití html či css. Jsou to různé uživatelské vstupy, icony, tabulky, drop-down menu a mnoho dalších.

5.7.4 Shinyjs

Rozšíření *shinyjs* je balík, který umožňuje využívat funkce javascriptu v prostředí Shiny. Jsou to funkce jako show, hide, toggle, hidden, disable, enable, reset a další. Výhoda spočívá v tom, že není potřeba znát javascript pro použití daných funkcí.

Je také možné vytvořit vlastní funkce v javascriptu a poté je volat s parametry v *server.R*.



Obrázek 8: Použití vlastní javascriptové knihovny v Shiny

5.7.5 shinyjqui

Je balík, který umožňuje využívat některé vlastnosti javascriptové knihovny jQuery v prostředí Shiny. Obsahuje některé funkce např. pro přesouvání, zmenšování, zvětšování, řazení objektů a další věci, které zvyšují interaktivitu v prostředí Shiny.

5.7.6 Shinydashboard

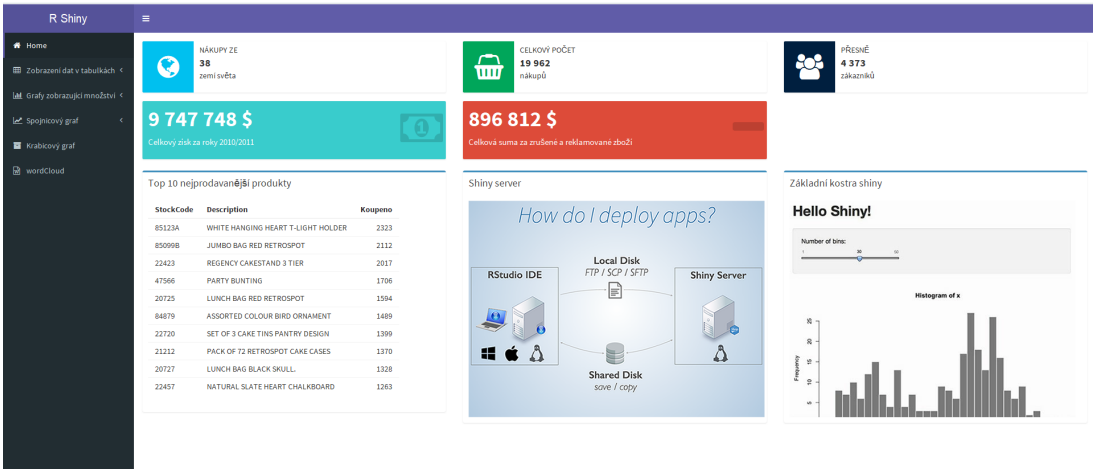
Je soubor vlastností pro vytváření dashboardů v prostředí Shiny. Shiny dashboard podporuje všechny funkce stejně jako v Shiny a navíc obsahuje některé navíc.

5.8 Celkové zhodnocení balíku shiny

Balík *shiny* je zajímavá eventualita pro vytváření webových aplikací pod R. Ohledně obtížnosti tvorby aplikací je poměrně důležité pochopit koncept reaktivního programování. Při pochopení zmíněného konceptu je tvorba aplikací v Shiny intuitivní, navíc tvůrci rstudio.com pěkně zpracovali návody pro začátečníky, včetně výukových videí.

Pokud by jsme se zabývali otázkou pro co Shiny využít ? Dostali bychom pravděpodobně odpověď, že Shiny slouží pro zobrazování vizualizovaných dat. Jelikož Shiny spolupracuje s mnoha balíky jako *ggplot2* nebo *plotly*, lze poměrně jednoduše vytvářet vizualizovaný obsah, zobrazovaný přes webové rozhraní. Navíc obsahuje také mnoho druhů ovládacích prvků, které dotvářejí interaktivitu aplikací. Můžeme v reálném čase pomocí těchto prvků filtrovat množiny dat a zobrazovat grafické výstupy na míru. Navíc s podporou technologií jako HTML nebo CSS či Javascriptu je možné Shiny aplikace ještě více přizpůsobovat potřebám vývojářů. Oblíbenost balíku *shiny* bude pravděpodobně stále narůstat s tím, jak roste počet vývojářů, využívající jazyka R. Podle webu `https://spectrum.ieee.org/computing/software/the-2017-top-programming-languages` se umístil jazyk R na konečném 6. místě mezi všemi jazyky na světě.

6 Vývoj ukázkové webové aplikace



Obrázek 9: Náhled ukázkového projektu

6.1 Stručný popis webové aplikace

Projekt je rozvržen jako webový dashboard, přičemž hlavním cílem bylo využít zvolený soubor dat a co nejlépe tato data vizualizovat, pomocí ať už integrovaných grafů nebo pomocí knihoven třetích stran.

Rozvržení jako dashboard bylo vybráno, aby byli eliminovány určité rušivé elementy jako dlouhé texty, a bylo tak možné se soustředit převážně na vizualizovaný obsah.

Stránka se skládá ze tří hlavních části headeru (hlavičky), menu a body (těla). Hlavička je poměrně jednoduchá, obsahuje pouze nadpis a tlačítko pro skytí menu. Menu obsahuje buďto obyčejné položky, na které lze jen kliknout a nebo pak sub položky (sub menu), kdy se po kliknutí rozevře menu a jsou dostupné další odkazy.

Body (tělo) slouží pro celkový obsah. Na úvodní stránce jsou info boxy a value boxy, které slouží pro zobrazení základních informací, které byli z dat získány. Při vytváření obsahu byly použity také různé uživatelské vstupy, které mají umožnit uživateli větší interakci s aplikací, lepší řazení a také výběr pouze určitých dat.

6.2 Základní struktura ukázkového projektu

Jako základní kostra ukázkové aplikace byli použity funkce obsažené v balíku Shiny Dashboard. Slouží pro vytvoření hlavičky, postranního panelu, kde je umístěno menu a tělo aplikace, určené pro obsah.

```
dashboardPage(
  dashboardHeader(),
  dashboardSidebar(),
  dashboardBody()
)
```

Výpis 4: základní struktura shiny dashboard

6.2.1 Menu

Menu je možné do aplikace přidat pomocí funkce *sidebarMenu()*, popřípadě je možné ho přidat i do headeru příkazem *dropdownMenu()*. V případě této ukázkové aplikace je umístěno v levém postranním panelu.

Položky v menu jsou rozděleny buď na položky, kde lze klasicky kliknout a zobrazí se daný obsah. Menu obsahuje další rozevírací sub menu s dalším obsahem.

```
sidebarMenu(
                menuItem("Home",tabName ="home",icon =icon("home")),
                menuItem("Zobrazeni dat v tabulkach",icon =icon("table"),
                menuSubItem("Klasicka tabulka",tabName="
                    zrusene_objednavky_tabulka")),
```

Výpis 5: Menu

6.2.2 Práce s CSS

V ukázkovém projektu byly využity také kaskádové styly. Bylo nutné změnit pozadí pro obsah, protože přednastavená barva příliš nevyhovovala po grafické stránce. Pro tyto potřeby byl do projektu přidán soubor *styly.css*. Jak už bylo zmíněno v podkapitole 5.6.1, tento soubor se musí nacházet v adresáři *www*, který přesně slouží pro tyto externí soubory. Samozřejmě by stačilo styl vložit do souboru uživatelského rozhraní, jenže nebylo předem jasné, kolik stylů bude. Navíc tento postup slouží hlavně pro ukázku. Následující ukázka demonstruje zadání cesty k externímu css souboru.

```
tags$head(
    tags$link(rel='stylesheet',type='text/css',href='styly.css')
```

Výpis 6: Nalinkování css souboru

Změnu pozadí pak lze provést:

```
.content-wrapper {
background-color: #ffffff;
}
```

Výpis 7: Změna pozadi na bílou barvu

6.3 Data

Pro tento ukázkový projekt byla vybrána veřejně přístupná kolekce dat, nad kterou jsou vytvořeny ukázkové vizualizace. Tato datová kolekce obsahuje data online obchodování mezi roky 1.12.2010 do 09.12.2011. Celkový počet záznamů je 541 909. Pro uložení a práci s těmito daty byla použita databáze MS-SQL.

Datová kolekce obsahuje následující atributy:

1. InvoiceNo: Číslo faktury může obsahovat na začátku písmeno 'c', které značí zrušení faktury.
2. StockCode: Kód produktu.
3. Description: Jméno, popřípadě popis produktu.
4. Quantity: Množství daného zboží.
5. InvoiceDate: Datum faktury.
6. UnitPrice: Cena za kus.
7. CustomerID: Identifikační číslo zákazníka.
8. Country: Země zákazníka.

6.3.1 Práce s odbc

Pro práci s ODBC a SQL serverem budeme potřebovat příslušný ovladač. Ovladač je dostupný na stránkách Microsoftu: `https://www.microsoft.com/en-us/download/details.aspx?id=53339`. Dále je nutné mít nainstalován balík *RODBC*, který obsahuje funkcionalitu pro přístup k odbc z prostředí R. Po instalaci je potřeba vytvořit připojení k SQL serveru pomocí ovladače ODBC. Kompletní nastavení bude dostupné v příloze.

Jakmile je připraveno spojení mezi ODBC a SQL serverem, můžeme k databázovému serveru přistoupit z prostředí R.

Následující ukázka demonstruje připojení k serveru. Prvním parametrem je jméno ODBC připojení. Následující dva parametry jsou uživatelské jméno a heslo k SQL serveru.

```
con<-odbcConnect("bpdb",uid ="uzivatel",pwd="heslo")
```

Výpis 8: pipe operator dplyr

Po úspěšném připojení k databázovému serveru můžeme zadat SQL dotaz a načíst data do prostředí R. Následující ukázka popisuje vytvoření dotazu pomocí funkce *sqlQuery* a také ukončení spojení se serverem pomocí *odbcClose*.

```
 data<-sqlQuery(con,"SELECT * FROM retail WHERE Country='Czech republic'")
odbcClose(con)
```

Výpis 9: Vytvorení SQL dotazu

6.3.2 Práce s balíkem dplyr

Při práci v prostředí R je často nutné manipulovat s daty. Data můžeme například filtrovat, třídit,seskupovat nebo vytvářet nové hodnoty z již existujících hodnot. Balík *dplyr* má přesně tuto funkcionalitu. Kromě toho *dplyr* využívá takzvaný pipe operátor. Je to typ operátoru, který zpřehledňuje zápis funkcí. V případě *dplyr* je to aplikace více funkcí na jeden data frame. V podstatě pipe operátor %>% vezme hodnotu z leva a předá ji jako argument doprava.

Na následující ukázce můžeme vidět využití pipe operátoru a také funkci *mutate*, která umožní vytvořit nový sloupec procenta. Nakonec jsou data setříděna funkcí *arrange*.

```
zruseneObjednavkyPocet<-zruseneObjednavkyPocet %>%
    mutate(procenta=Pocet/(sum(Pocet)/100)) %>%
    arrange(-procenta)
```

Výpis 10: pipe operator a funkce balíku dplyr

6.4 Vizualizování dat a celkové dotváření obsahu

6.4.1 Tabulka

Zobrazit data pomocí tabulky je jeden z přístupů, jak data vizualizovat v Shiny. Při vytváření tabulky, ale i jiných objektů třeba grafů se dodržuje následující postup:

V části aplikace pro uživatelské rozhraní *ui.R* definujeme výstup, kam se má tabulka vyrenderovat. Jak bylo zmíněno v podkapitole 5.5.2, každá Render funkce slouží pro vyrenderování něčeho jiného (není univerzální). Pro tabulku definujeme:

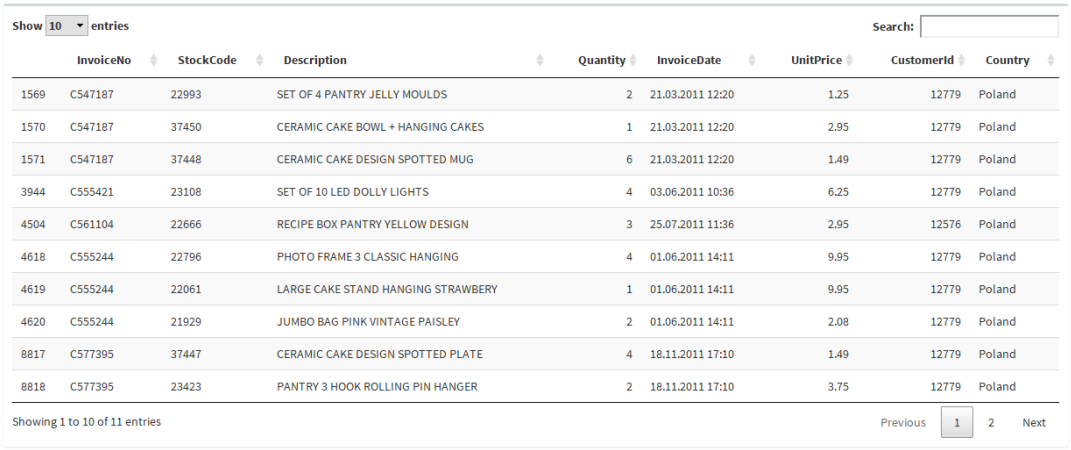
```
dataTableOutput("tabData")
```

Výpis 11: Definování výstupu pro tabulku

Následně v serverové části aplikace *server.R* využijeme tento výstup a použijeme funkce pro renderování tabulky. Jako zdroj dat je očekáván datový rámec *data frame*.

```
output$tabData<-DT::renderDataTable( { })
```

Výpis 12: Renderování tabulky



Obrázek 10: Ukázka tabulky

6.4.2 Vytváření koláčového grafu graphics

V podkapitole 4.4 bylo zmíněno, že software R obsahuje již několik zabudovaných grafů, prostřednictvím balíku *graphics*. K praktické ukázce byl vybrán jeden typ z této knihovny a to koláčový graf. Tento typ grafu se používá pro znázornění podílu dílčích hodnot na celku. Koláčový graf lze vytvořit funkcí *pie()*, kde je očekáván jako parametr vektor. Ostatní parametry jsou již dobrovolné:

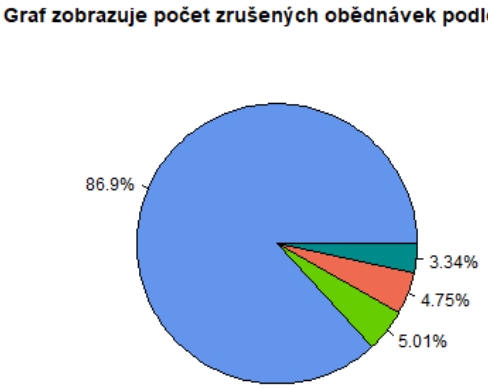
1. labels: parametr pro přidání textových popisků jednotlivých výsečí.
2. col: parametr pro specifikaci barevnosti grafu.
3. main: nastavení nadpisu grafu.

Pro vytvoření grafu opět musíme definovat výstup v *ui.R*. Pro grafy je to *plotOutput()*, ve kterém opět musíme definovat *outputId*. V případě ukázky 6.4.2 je *outputId kolacovyGraf*.

```
output$kolacovyGraf<-renderPlot({
pie(zruseneObjednavkyPocet$procenta,labels =paste0(
    zruseneObjednavkyPocet$procenta,"%"),col=colors,main="Graf zobrazuje pocet
    zrusenych obednavek podle zemi")
})
```

Výpis 13: Vyrenderování koláčového grafu

Kromě samotného grafu je možné přidat také legendu. Slouží k tomu metoda *legend()*, kde je nutné specifikovat, kde má být legenda umístěna a jaké popisky má zobrazovat.



Obrázek 11: Ukázka výsečového grafu

6.4.3 Grafy z ggplot2

Jako další ukázkový graf byl vybrán graf sloupcového typu.

V ukázkové aplikaci byl tento typ grafu použit k zobrazení počtu zrušených objednávek v závislosti na zemi, kde byla objednávka vytvořena.

Pro vytvoření tohoto grafu byla využita knihovna *ggplot2*. Klíčovou vlastností následujícího kódu je *geom_bar*, touto vlastností udáváme, že graf bude sloupcového typu. Vlastností *geom_text* se nastaví zobrazení hodnot nad jednotlivými sloupci v grafu. Pomocí *ggtitle* je možné zobrazit libovolný nadpis, přičemž pomocí *theme* můžeme stylovat celkové zobrazení grafu, jako nastavení barvy písma, velikost písma, typu písma a další.

```
ggplot(data=data,aes(x=Country,y=Pocet,fill=Country),width="600px")+geom_bar(
    stat ="identity") +
    geom_text(aes(label=Pocet),position =position_dodge(width =0.9 ),vjust
        =-0.25 ) +
    ggtitle("Graf zobrazuje pocet zrusenych obednavek podle zemi") +
    theme(plot.title = element_text(color="black", size=15, face="bold.italic")
        )
```

Výpis 14: Vytvoření grafu pomocí ggplot2

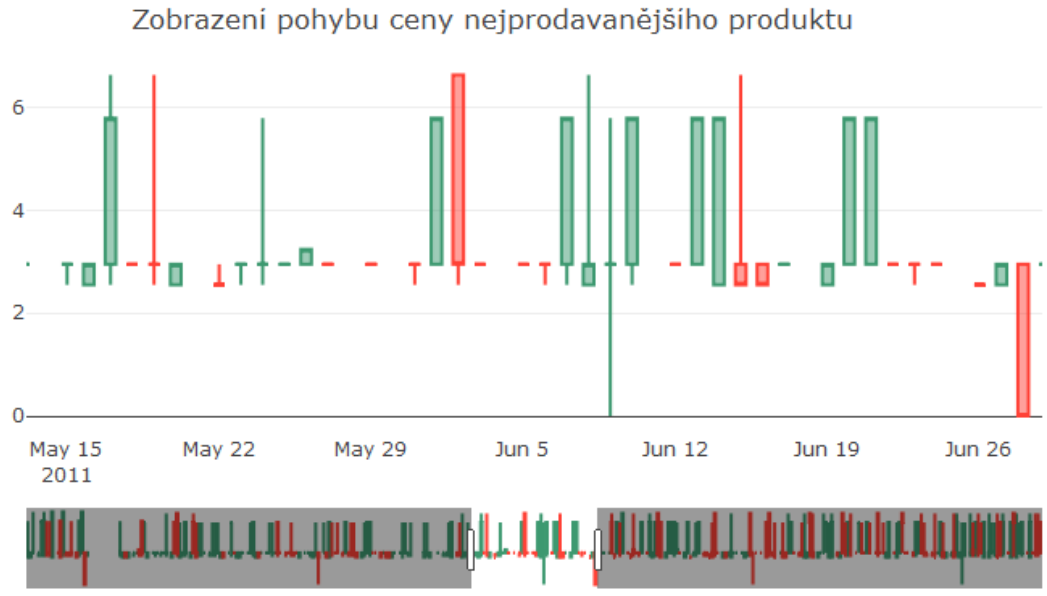
6.4.4 Svíčkový graf

Svíčkové grafy patří mezi nejpoužívanější typ grafu v obchodní sféře. Využívají se pro předpověď budoucího směru trhu.

Základem je svíčka, ze které mužeme vyčíst čtyři hodnoty (open, close, max, min), přičemž tyto svíčky jsou zobrazovány vždy v nějakém časovém období.

Pro vytvoření svíčkového grafu bylo potřeba využít v ukázkové aplikaci balíku *plotly*. Výhoda *plotly* je v zabudovaných interaktivních funkcích jako zoomování, posouvání grafu, popřípadě možnost si graf uložit.

Jedna z dalších výhod je responzivita - graf se dobře přizpůsobuje jakémukoli rozlišení obrazovky.



Obrázek 12: Svíčkový graf z balíku plotly

Grafy v plotly jsou rozdílné i co se týče vyrenderování grafu. Neslouží k tomu funkce *shiny*, ale *plotly* obsahuje vlastní funkci:

```
output$candlestick_plot<-renderPlotly({
```

Výpis 15: funkce pro renderování v plotly

Pro definování výstupu v *ui.R* slouží funkce:

```
plotlyOutput("candlestick_plot")
```

Výpis 16: definování výstupu pro graf ui.R (plotly)

Následující 6.4.4 ukazuje strukturu vytvoření svíčkového grafu v plotly. Pro vytvoření grafu je potřeba definovat datový rámec, v tomto případě topProduct následuje funkce *plot_ly*, kde pomocí parametru type definujeme typ grafu.

```
topProduct %>%
    plot_ly(x = ~date, type="candlestick",
            open = ~open, close = ~close,
            high = ~max, low = ~min)
```

Výpis 17: Vytvoření svíčkového grafu v plotly

6.5 Mrak slov (Word cloud)

Mrak Slov nebo také (Word cloud, Tag cloud) je jeden z druhů vizuálního zobrazení, obvykle některých klíčových slov na webových stránkách. Značky jsou obvykle jednotlivá slova, přičemž velikost slov v mraku zavisí na počtu, kolikrát bylo slovo obsaženo v textu, ze kterého byly slova získány. Slova jsou seřazeny nejčastěji do kruhu, ale i do jiných útvarů.

Pro lepší přehlednost mohou být slova také barevně odlišena.

Shiny aplikace umožňuje použít i tento typ vizualizace. V ukázkové aplikaci sloužil Mrak slov pro vizualizaci textových popisků produktu (databázový atribut Description). Pro tento typ vizualizace je potřeba použít balík wordcloud, přičemž pro získání dat z textu slouží balík tm.

Aby bylo možné slova správně zobrazit, bylo nutné odfiltrovat některé části textu jako předložky, spojky, čísla, speciální znaky a slova, které nemají význam. Balík tm obsahuje funkce pro práci s textem a text mining. Navíc má integrovanou databázi (anglických) slov, které nemají dostatečný význam.

Pro odstranění těchto slov je připravena funkce *removeWords()*. Parametrem je daný textový řetězec. Druhý parametr je seznam řetězců pro odstranění - *stopwords()*, což je databáze slov.

```
dta<-removeWords(dta,stopwords())
```

Výpis 18: Funkce pro odstranění nedůležitých slov

Dále je zde funkce pro odstranění určitých vzorů a nahrazení těchto znaků něčím jiným, například mezerou. Pomocí *gsub()* byli odstraněny znaky od (A-z).

Rovněž nebylo nutné mít některá slova s velkým počátečním písmenem a některá s malým (jelikož je to stále jedno slovo). Proto byla využita funkce *tolower()*, pro nastavení všech slov malými písmeny.

```
dta<-tolower(dta)
dta<-gsub(pattern ="\\b[A-z]\\b{1}",replace=" ", dta)
```

Výpis 19: Funkce pro odstranění nedůležitých slov

Při těchto operacích s textem vznikají dlouhé mezery mezi slovy. Dají se odstranit funkcí *stripWhitespace()*. Jakmile je text připraven, stačí použít funkci *wordcloud()*, kde je také parametr *min.freq*, který udává od jakého počtu slov, nalézajícím se v textu, má být slovo zobrazeno.

Pro ukázkovou aplikaci byl jako parametr použit *slider*, který k tomuto účelu vyhovovoval. Je tedy možné nastavit počet zobrazujících se slov uživatelem.

```
wordcloud(dta,min.freq =input$slider_worldCloud)
```

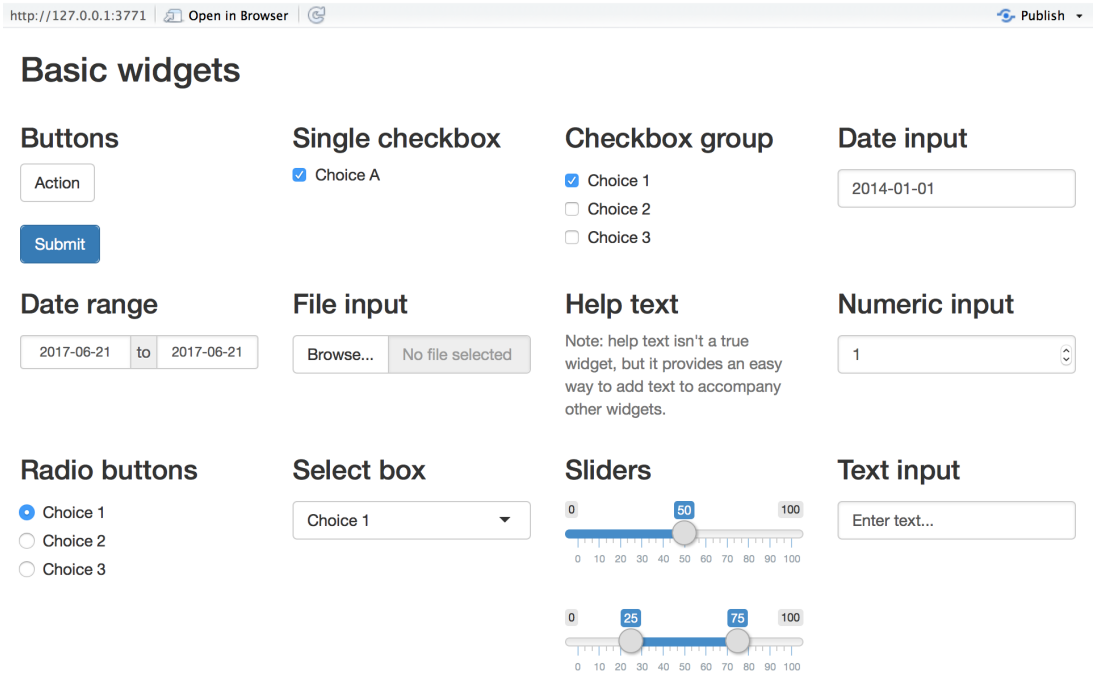
Výpis 20: Funkce pro odstranění nedůležitých slov

Obrázek 13: Ukázka WordCloudu

6.6 Ovládací prvky

Kromě vytváření grafů je potřeba reagovat také na vstupy uživatele. Uživatel může například přepínat mezi typem grafů nebo pouze filtrovat, které záznamy chce vidět.

Základní funkcionalita pro vytvoření uživatelských vstupů je součástí balíku shiny. Seznam všech uživatelských vstupů je na následujícím obrázku 14.



Obrázek 14: Kompletní přehled uživatelských vstupu z balíku shiny

6.6.1 Ovládací prvek Selectize a čtení ze souboru

V praktické části bakalářské práce bylo pro ukázku také implementováno čtení ze souboru. Pro větší interaktivitu je možné vybírat sloupce načtené ze souboru a ty pak zobrazit na krabicovém grafu. Cílem bylo ukázat použití reaktivních výrazů.

Shiny obsahuje pro zadání cesty k souboru ovládací prvek *fileInput()*. Pro výběr prvků ze souboru slouží *selectizeInput()*. V této ukázce je využit reaktivní výraz *observe*. *Observe* může číst reaktivní hodnoty a volat reaktivní výrazy, ale na rozdíl od ostatních výrazů, nemůže poskytovat výsledek a nemůže sloužit jako vstup jiných reaktivních výrazů. Zde *observe* slouží pro aktualizaci prvků v *selectizeInput()*.

```
observe({

    updateSelectizeInput(session,"file_columns",choices =names(fileCsv))

})
```

Výpis 21: Ukázka observe

6.7 Komplikace

Při vytváření tohoto ukázkového projektu jsem narazil na jeden problém. Tento problém mohl ovlivnit vzhled stránky i samotnou funkčnost. Pro vytvoření vizualizace dat pomocí tabulky, jsem v prvé řadě použil dostupnou funkcionalitu přímo z balíku shiny. Při testování jsem zkoušel, jak se jednotlivé elementy stránky chovají i v jiných rozlišeních. Zjistil jsem, že při rozlišení 800x600 a menším se tabulka špatně přizpůsobuje rozlišení, kdy některé sloupce se nevykreslovali , nebo se celá tabulka centrovala na pravou stranu. Proto jsem použil externí knihovnu *DT*, kde existuje parametr *ScrollX*, který umožňuje při menších rozlišeních použít posuvník a tabulku tak lze číst na jakémkoli rozlišení.

7 Nasazení aplikace

Zde se vývoj aplikace dostává do fáze nasazení. K publikování Shiny projektů slouží Shiny Server, o kterém se zmiňuje obecně podkapitola 5.3. Ukázková aplikace byla nasazena na virtuální server s operačním systémem Ubuntu 16.4. Ještě před samotnou instalací softwaru Shiny Server je nutné mít nainstalován software R a také balík *shiny*. Popis instalace včetně konfigurace serveru na jednotlivé operační systémy, můžeme najít na adrese: `http://docs.rstudio.com/shiny-server/`. Je nutné doinstalovat také všechny externí balíky, které jsou v projektu využity. Pro správnou funkčnost by balíky měli být uloženy v */usr/local/lib/R/site-librarary*. Dále je potřeba doinstalovat ODBC ovladač pro Ubuntu. Odkaz na stažení můžeme najít v podkapitole 6.3.1. Pro konfiguraci ODBC připojení je klíčový soubor *odbc.ini*, nacházející se v adresáři *etc*. Soubor má nasledující strukturu:

```
//Jmeno odbc
[bp_db]

//Pouzivany ovladac
Driver = SQL Server Native Client 11.0

//Nazev database
Database   =

//SQL-Server
Server     =
```

Výpis 22: Struktura souboru odbc.ini

Jakmile je vše připraveno, můžeme aplikaci nasadit na server. Shiny aplikace se nalézají v adresáři */srv/shiny-server/*. Ukázková aplikace byla uložena do adresáře *sample-apps/hello*. Pokud chceme k aplikaci přistoupit z prohlížeče je nutné zadat také port. Shiny aplikace jsou dostupné na portu :3838 přičemž přístup k aplikaci by byl následující: http://ip_adresa_serveru:3838/sample-apss/hello

8 Závěr

Cílem mé práce bylo vytvořit ukázkový projekt, který byl postaven na frameworku Shiny. Pro tvorbu zmíněného projektu byl využit programovací jazyk R. Za účelem vytvoření přehledného webu, primárně určeného pro grafickou vizualizaci dat, byl využit balík *shiny dashboard* jako základní kostra webu. V teoretické části jsem zpracoval potřebné informace, týkající se pojmů Big data a některých nástrojů a technologií, které s tímto tématem souvisí. Výsledkem této bakalářské práce je funkční, webová aplikace, která demonstruje vizualizaci dat, převážně pomocí grafů. V práci je také ukázáno využití HTML a CSS, umožňující s Shiny spolupracovat. V neposlední řadě je popsáno využití externích balíků (knihoven), které byli v práci využity. Celou práci lze zhlédnout na adrese `http://vps474844.ovh.net:3838/sample-apps/hello/`

Veškeré obrázky byly převzaty z veřejně dostupných zdrojů a slouží pouze pro účely tvorby této bakalářské práce.

Literatura

[1] *cisco.com* [online]. [cit. 3. 3. 2018]. Dostupné z: <https://www.cisco.com/c/cs_cz/about/news/2015/20150608.html>

[2] HOLUBOVÁ, Irena, Jiří KOSEK, Karel MINAŘÍK a David NOVÁK. *Big Data a NoSQL databáze*. Praha: Grada, 2015. Profesionál. ISBN 978-80-247-5466-6.

[3] *Big Data. In: ManagementMania.com* [online]. [cit. 2. 4. 2018]. Dostupné z: <https://managementmania.com/cs/big-data>

[4] *r-project.org* [online]. [cit. 4.3. 2018]. Dostupné z: <https://www.r-project.org/about.html>

[5] *fu.ff.cuni.cz* [online]. [cit. 22. 3. 2018]. Dostupné z: <http://fu.ff.cuni.cz/PROG/prog10reaktivni.html>

[6] *shiny.rstudio.com* [online]. [cit. 22. 3. 2018]. Dostupné z: <https://shiny.rstudio.com/articles>

[7] Beeley, Chris. *Web Aplication Development with R using Shiny*. Birmingham. 2013. ISBN 978-1-78328-447-4.

[8] *docs.mongodb.com* [online]. [cit. 25. 3. 2018]. Dostupné z: <https://docs.mongodb.com/>

[9] *storpool.com* [online]. [cit. 17. 4. 2018]. Dostupné z: <https://storpool.com/blog/what-is-distributed-storage-system>

[10] *chip.cz* [online]. [cit. 20. 4. 2018]. Dostupné z: <https://www.chip.cz/novinky/novinka-od-oracle-databaze-in-memory/>

[11] *guides.library.duke.edu* [online]. [cit. 2. 4. 2018]. Dostupné z: <https://guides.library.duke.edu/datavis/vis_types>

[12] *zdrojak.cz* [online]. [cit. 20. 4. 2018]. Dostupné z: <https://www.zdrojak.cz/clanky/navrh-databaze-nosql-vs-sql/>

[13] *vtm.e15.cz* [online]. [cit. 8. 4. 2018]. Dostupné z: <http://vtm.e15.cz/aktuality/data-mining-jiny-pohled-na-problem>

[14] *ceskatelevize.cz* [online]. [cit. 8. 4. 2018]. Dostupné z: <http://www.ceskatelevize.cz/porady/10121359557-port/tema/125-datamining-dolovani-dat/>

[15] *businessit.cz* [online]. [cit. 12. 4. 2018]. Dostupné z: <http://www.businessit.cz/cz/systemy-pro-prediktivni-analyzu-pomohou-resit-i-vase-problemy.php>

[16] *redis.io* [online]. [cit. 9. 4. 2018]. Dostupné z: <https://redis.io/documentation>

[17] *objevit.cz* [online]. [cit. 9. 4. 2018]. Dostupné z: <http://objevit.cz/neni-vsechno-zlato-co-se-trpyti-vizualizace-dat-t35829>

[18] *techopedia.com* [online]. [cit. 9. 4. 2018]. Dostupné z: <https://www.techopedia.com/definition/4517/open-database-connectivity-odbc>

[19] MURRELL, Paul. *R graphics. 2nd ed.* Boca Raton: CRC Press, c2011. ISBN 978-1439831762.

A Seznam příloh

Obsah přiloženého DVD
\webovaAplikace - kompletní zdrojové kódy aplikace
\soubor.csv - soubor pro otestování čtení ze souboru ve formatu CSV
\data.txt - soubor obsahující odkaz na veřejně dostupnou kolekci dat
\JIR0082.pdf - textová část bakalářské práce ve formátu PDF